Brief Announcement: Temporal Locality in Online Algorithms

Maciej Pacut
Technical University of Berlin, Germany

Mahmoud Parham
Faculty of Computer Science, University of Vienna, Austria

Joel Rybicki
Institute of Science and Technology Austria, Austria

Stefan Schmid
TU Berlin & Fraunhofer SIT, Germany

Jukka Suomela
Aalto University, Finland

Aleksandr Tereshchenko
Aalto University, Finland

Abstract

Online algorithms make decisions based on past inputs, with the goal of being competitive against an algorithm that sees also future inputs. In this work, we introduce *time-local online algorithms*; these are online algorithms in which the output at any given time is a function of only T latest inputs. Our main observation is that time-local online algorithms are closely connected to *local distributed graph algorithms*: distributed algorithms make decisions based on the *local information in the spatial dimension*, while time-local online algorithms make decisions based on the *local information in the temporal dimension*. We formalize this connection, and show how we can directly use the tools developed to study distributed approximability of graph optimization problems to prove upper and lower bounds on the competitive ratio achieved with time-local online algorithms. Moreover, we show how to use *computational techniques* to synthesize optimal time-local algorithms.

2012 ACM Subject Classification Theory of computation → Online algorithms; Theory of computation → Distributed computing models

Keywords and phrases Online algorithms, distributed algorithms

Digital Object Identifier 10.4230/LIPIcs.DISC.2022.41

Category Brief Announcement

Related Version *Full Version*: https://arxiv.org/abs/2102.09413

Funding This research has received funding from the German Research Foundation (DFG), grant 470029389 (FlexNets), 2021-2024, and the Marie Skłodowska-Curie grant agreement No. 840605.

1 Introduction

A common setting in theoretical computer science is that there is a sequence of n inputs and we need to produce a sequence of n outputs. In the case of classic centralized algorithms, each output may arbitrarily depend on any part of the input. However, there are two key settings in which outputs are produced based on *partial* inputs (see Figure 1):

- In distributed computing, we can interpret the input sequence as a path formed by n computers; each computer holds a local input and each computer has to produce a local output. In this setting, fast distributed algorithms are also local: if the algorithm stops

36th International Symposium on Distributed Computing (DISC 2022).
Editor: Christian Scheideler; Article No. 41; pp. 41:1–41:3
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

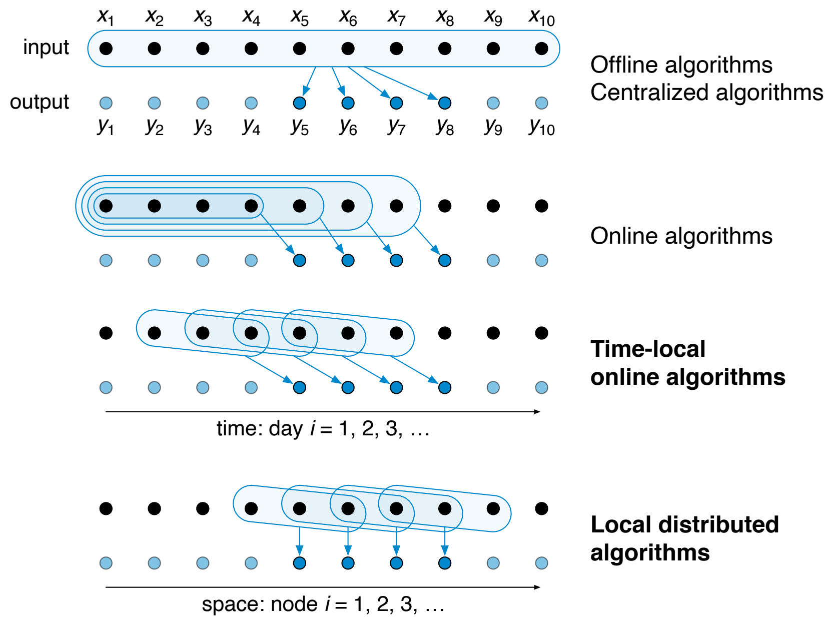

Figure 1 Local decision-making in time vs. space dimensions.

after $T = O(1)$ communication rounds, then the output of computer number i only depends on the inputs of computers $i - T, \ldots, i + T$.

- In online algorithms [3], we can interpret the input sequence as a time series, and the output sequence as a sequence of decisions. At each time point i we need to make a decision that is based on past inputs. That is, output at time point i only depends on the inputs at time points $1, \ldots, i - 1$.

While these two settings share the feature that each output value depends only on some but not all input values, this connection does not seem to enable much technology transfer between the two domains. In particular, online algorithms are fundamentally infinite objects, as the output may depend on an unbounded number of previous inputs.

In this work, we introduce *time-local online algorithms*; these are online algorithms in which the output at any given time is a function of only T latest inputs, instead of the full history of past inputs. Such algorithms (1) have new attractive properties that are not exhibited by general online algorithms and (2) have many similarities with local distributed graph algorithms, enabling one to transfer tools and techniques between the two domains.

2 Benefits of Time-Local Online Algorithms

Fault-Tolerant Distributed Decision. Time-local online algorithms lead to fault-tolerant distributed decision-making. Consider a setting in which many geographically distributed computers need to make *consistent* decisions. All computers can observe the same input stream, and each day each of them has to announce its own decision.

If all computers are started at the same time, we can take any deterministic online algorithm and let each computer run its own copy of the algorithm. However, this approach does not *tolerate failures*: if a computer crashes and is restarted, the local state of the algorithm is lost, and as the decisions may depend in general on the full history of inputs, it will no longer make consistent decisions with the others.

Deterministic time-local online algorithms automatically guarantee that all computers will make consistent decisions. The system will tolerate an arbitrary number of failures and ensure that the computers will also recover from transient faults, i.e., it is *self-stabilizing* [4]: in T steps since the latest failure, all computers will deterministically make consistent decisions, without any communication.

Random Access to the Decision History. Time-local online algorithms make it possible to efficiently access any past decision with zero additional storage beyond the storage of the input stream. To recover a past decision at any time i, it is sufficient to look up the last T inputs at time i and apply the deterministic time-local algorithm.

3 Connection with Distributed Computing

As illustrated in Figure 1, time-local online algorithms are very similar to local distributed algorithms in directed paths: distributed algorithms make decisions based on the *local information in the spatial dimension*, while time-local online algorithms make decisions based on the *local information in the temporal dimension*. One key difference is that time-local online algorithms are *one-sided*—output i depends only on previous T inputs—while local distributed algorithms are *two-sided*—output i can depend on T inputs in either direction. However, it is easy to navigate between these two settings.

We consider two variants of time-local online algorithms. *Unclocked* algorithms make a decision at time i without knowing the value of i. Such algorithms very similar to deterministic distributed algorithms in the *port-numbering model* [1]—in particular, we face the same challenge of local symmetry breaking. *Clocked* time-local online algorithms can depend on the value of i. Such algorithms turn out to be similar to deterministic distributed algorithms in the *supported* LOCAL model [5]. The key difference is that clocked time-local online algorithms do not know the length of the input sequence in advance, while in the supported LOCAL model the input size is also known.

4 Algorithm Synthesis

In the full version of this work, we describe an *algorithm synthesis method* that one can use to design optimal time-local online algorithms for small values of T, for problems with finite input and output domains. We demonstrate the power of the technique in the context of a variant of the *online file migration problem* [2], and show that e.g. for two nodes and unit migration costs there exists a 3-competitive time-local algorithm with horizon $T = 4$, while no deterministic online algorithm (in the classic sense) can do better.

References

1. Dana Angluin. Local and global properties in networks of processors. In *Proc. 12th Annual ACM Symposium on Theory of Computing (STOC 1980)*, 1980. doi:10.1145/800141.804655.
2. Marcin Bienkowski. Migrating and replicating data in networks. *Computer Science-Research and Development*, 27(3):169–179, 2012.
3. Allan Borodin and Ran El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, 1998.
4. Shlomi Dolev. *Self-Stabilization*. MIT Press, 2000.
5. Klaus-Tycho Foerster, Juho Hirvonen, Jukka Suomela, and Stefan Schmid. On the power of preprocessing in decentralized network optimization. In *Proc. 28th IEEE Conference on Computer Communications (INFOCOM 2019)*, 2019. doi:10.1109/INFOCOM.2019.8737382.